IN THE CLAIMS

For the convenience of the Examiner, all pending claims of the Application are reproduced below.

1. **(Currently Amended)** A computer implemented method for providing system security and resource management, comprising:

managing event messages by a master control processor between system handlers according to security system policies;

processing network messages by a network handler between client and server computers;

inserting native and third party event messages received by an insertion handler into the master control processor for processing by other system handlers;

detecting and processing event message signatures by ~~**the signature**~~ **a signature** handler from alarm, system, and insertion events for conversion into system alarm messages for action by the other system handlers; and

performing actions by an action handler in response to action requests **generated by the other system handlers and received** from the master control processor**, wherein the master control processor routes messages to and from the system handlers without being involved in security and resource management functions distributed among the system handlers**.

2. **(Original)** The method according to claim 1, further comprising maintaining an execution environment by an agent handler for mobile autonomous code modules.

3. **(Original)** The method according to claim 1, further comprising collecting and logging event messages by a logging handler.

4. **(Original)** The method according to claim 1, further comprising managing system configuration parameters by a configuration handler.

5. **(Currently Amended)** The method according to claim 1, wherein the step of managing event messages comprises:

registering the system handlers;

passing event messages between system handlers; and

managing ~~a event queues~~ **an event queue** attached to the system handlers.

6. **(Currently Amended)** The method according to claim 5, wherein the step of registering each system handler comprises:

reading **a module of** the **system** handler ~~module~~ to determine initialization requirement;

initializing **an application programming interface of** the **system** handler ~~application programming interface~~;

determining if the **system** handler is to run as a remote procedure call;

making the **system** handler available through **an interface of** the remote procedure call ~~interface~~ if run as ~~a remote~~ **the remote** procedure call; and

initializing ~~the handler~~ input/output queues **of the system handler** if not run as ~~a remote~~ **the remote** procedure call.

7. **(Original)** The method according to claim 1, wherein the system handlers comprise static and dynamic system processes.

8. **(Original)** The method according to claim 1, further comprising;

initiating the system handlers by internal mechanisms;

initiating the system handlers by external mechanisms; and

initiating the system handlers from the master control processor.

9. **(Currently Amended)** The method according to claim 1, ~~wherein the system handlers have a reversible architecture to enable the~~ **further comprising operating the** system handlers ~~to be used~~ in either a client or server computer mode.

10. **(Currently Amended)** The method according to claim 1, wherein the step of processing network messages by the network handler comprises:

allowing connection only from clients and servers defined in ~~a access~~ **an access** control list;

authenticating protocols with clients and servers;

compressing data to minimize bandwidth requirements; and

encrypting data to provide secure communication.

11. **(Currently Amended)** The method according to claim 1, wherein the step of inserting native and third party event messages received by the insertion handler from external programs comprises:

reading and writing messages using an insertion method selected from ~~the group~~ **a group** consisting of file descriptor, network sockets and named pipe;

using a native insertion mode library to directly insert messages into the master control processor; and

using an external insertion mode library linked to a third party ~~sources~~ **source** to directly insert messages into the master control processor.

12. **(Currently Amended)** The method according to claim 1, wherein the step of detecting and processing event messages received by the signature handler comprises:

accepting alarm events from the master control processor;

decoding ~~the alarm~~ **an alarm** type and originator from the alarm event;

consulting internal signature registry for alarms of the type accepted;

handing the alarm message off to ~~the signature~~ **a signature** module for processing;

extracting alarm data macro information;

determining if an alarm has occurred;

consulting the action policy if an alarm has occurred to determine response; and

passing the resulting response message to the master control processor for action by the other system handlers.

13. **(Currently Amended)** The method according to claim 1, wherein the action performed by the action handler is selected from ~~**the group**~~ **a group** consisting of blocking a host with a modified route command, blocking a host with a packet filter modification command, disabling a user account, disabling a network interface, running an external user-defined command, logging an event, sending email or pager alerts, sending on-screen alerts to users or administrators, requesting server executed action, and a pluggable action defined by a user.

14. **(Currently Amended)** The method according to claim 1, wherein the steps of detecting and processing event messages received by the signature handler and performing actions by the action handler comprises the steps of:

receiving an event message containing a signature by the signature handler from the master control processor;

detecting an attack by the signature handler from the event message signature;

consulting action policy by the signature handler;

determining action to be taken by the signature handler;

ending the ~~**process**~~ **processing** if no action is required;

determining required action parameters by the signature handler if action is required;

sending an action request to the action handler by the signature handler via the master control processor;

processing and executing the action request by the action handler; and

returning status of the executed action to the signature handler.

15. **(Original)** The method according to claim 2, wherein the step of maintaining an execution environment by the agent handler for mobile autonomous code modules comprises:

ensuring that the mobile autonomous code modules carry appropriate credentials, are authenticated and cryptographically signed by a trusted introducer, and able to execute on the host operating system;

distributing mobile autonomous code modules to one or more client computers;

executing the mobile autonomous code modules without interference;

allowing the mobile autonomous code modules to collect and report its results; and

shutting down the mobile autonomous code modules.

16. **(Original)** The method according to claim 15, wherein the step of executing the mobile autonomous code modules comprises:

verifying detected alarms;

reducing false alarm rates; and

providing immediate response.

17. **(Original)** The method according to claim 15, wherein the step of executing the mobile autonomous code modules comprises the steps of actively looking for problems and identifying attackers when problems are detected.

18. **(Original)** The method according to claim 15, wherein the step of executing the mobile autonomous code modules comprises performing security and system administration tasks in self-healing network environments.

19. **(Currently Amended)** The method according to claim 15, wherein the step of executing the mobile autonomous code modules comprises allowing self-healing components ~~**of the system**~~ to move between clients and operate independently where required.

20. **(Original)** The method according to claim 2, wherein the step of maintaining an execution environment by the agent handler for mobile autonomous code modules comprises:

enabling self-healing and adaptive networks;

facilitating distribution of updates for the mobile autonomous code modules; and

centralizing command and control functions for increased reliability.

21. **(Original)** The method according to claim 2, wherein the step of maintaining an execution environment by the agent handler further comprises forming a peer-to-peer defensive cluster with mobile autonomous code modules.

22. **(Original)** The method according to claim 2, wherein the step of maintaining an execution environment by the agent handler further comprises protecting the mobile autonomous code modules from alteration or tampering by hostile adversaries, and dispatching the mobile autonomous code modules through a predictable schedule from a central control point.

23. **(Original)** The method of claim 22, wherein the mobile autonomous code modules are dispatched through a random schedule.

24. **(Original)** The method according to claim 2, wherein the step of maintaining an execution environment by the agent handler further comprises:

programming the mobile autonomous code modules to detect and remove attackers at random;

storing code for the mobile autonomous code modules at a central location;

preventing alteration of the mobile autonomous code modules on client computers;

updating the mobile autonomous code modules with updated security detection strategies without modifying client computers;

beginning an active search for attackers when alerted to an intruder's presence;

performing automated corrective measures to remove the intruder; and

saving forensic evidence.

25. **(Currently Amended)** The method according to claim 2, wherein security-specific mobile autonomous code modules are selected from ~~**the group**~~ **a group** consisting of forensic evidence agent, intrusion control agent, file integrity agent, host scanning agent, known intrusion agent, loadable kernel module agent, password cracking agent, log archive agent, rootkit agent, suspicious file agent, promiscuous mode agent, hidden process detection agent, unauthorized network daemon agent, self-test agent, spy agent, zombie shells agent, and insider attack agent.

26. **(Original)** The method according to claim 25, wherein forensic evidence gathered by the forensic evidence agent from protected systems is cryptographically signed to prevent tampering.

27. **(Currently Amended)** The method according to claim 2, wherein network management-specific mobile autonomous code modules are selected from ~~**the group**~~ **a group** consisting of backup agent, host inventory agent, system monitor and status agent, system task agent, and PPatchWatch™ ~~**agent**~~ **agent.**

28. **(Currently Amended)** The method according to claim 3, wherein the method of logging event messages is selected from ~~**the group**~~ **a group** consisting of text-based files, a local system auditing facility, a cryptographically signed secure log format, directly to a system console, e-mail notification, direct used interface, and system wide notification.

29. **(Original)** The method according to claim 4, wherein the step of managing system configuration parameters comprises;

interfacing generic calls to other system handlers;

reading system configuration parameters;

writing system configuration parameters;

changing system configuration parameters;

reverting system configuration parameters back to a previous version;

deleting system configuration parameters;

backing up system configuration parameters; and

providing multiple access protection mechanisms.

30. **(Currently Amended)** A system for providing system security and resource management, comprising:

a master control processor for managing event messages between system handlers according to security system policies;

a network handler for processing network messages between client and server computers;

an insertion handler for inserting native and third party event messages into the master control processor for processing by other system handlers;

a signature handler for detecting and processing event messages from alarm, system, and insertion events for conversion into system alarm messages for action by other system handlers; and

an action handler for performing actions in response to action requests from the master control processor**, wherein the master control processor routes messages to and from the system handlers without being involved in security and resource management functions distributed among the system handlers**.

31. **(Original)** The system according to claim 30, further comprising:

an agent handler for maintaining an execution environment for mobile autonomous code modules;

a logging handler for collecting and logging event messages; and

a configuration handler for managing system configuration parameters.

32. **(Currently Amended)** The system according to claim 31, wherein the system handlers **are operable to** auto-register themselves and their capabilities with the master control processor.

33. **(Original)** The system according to claim 30, wherein the action handler utilizes pluggable action modules.

34. **(Original)** The system according to claim 30, wherein the signature handler utilizes pluggable signature modules.

35. **(Currently Amended)** The system according to claim 34, wherein the pluggable signature modules **are operable to** auto-register with the signature handler.

36. **(Currently Amended)** The system according to claim 34, wherein ~~a pluggable~~ **the pluggable** signature modules ~~uses~~ **are operable to use** stored data from other signature modules, ~~stores~~ **operable to store** data between execution, ~~stores~~ **operable to store** data between system startup and shutdown sequences, and **operable to** only ~~processes~~ **process** signatures relating to the **pluggable** signature module.

37. **(Original)** The system according to claim 34, wherein the pluggable signature modules are added and removed from the system without modifying the core system code.

38. **(Original)** The system according to claim 31, wherein the system is installed on at least one server computer and at least one client computer.

39. **(Currently Amended)** The system according to claim 38, wherein the system ~~installed~~ **is installed** on at least one client ~~operates~~ **computer and is configured to operate** independently of the system installed on at least one server **computer** for reduced processing and reaction time and when network communications are disrupted.

40. **(Original)** The system according to claim 38, further comprising at least one graphical user interface.

41. **(Original)** The system according to claim 40, further comprising multiple levels of independent alarm filters on the client and server computers for reduced false alarm reporting, configuration flexibility, and system granularity.

42. **(Original)** The system according to claim 40, further comprising encrypted and mutually authenticated communication links between the server computer, client computer and graphical user interface.

43. **(Original)** The system according to claim 40, further comprising a means for re-allocating system resources to circumvent system problems or failures.

44. **(Currently Amended)** The system according to claim 31, wherein the system is installed on a client computer **and includes means operable** to maintain host processes, collect and forward events, process local client signatures, generate events, initiate and respond to action requests, initiate self-healing counter-measures, and host mobile autonomous code agents.

45. **(Currently Amended)** The system according to claim 31, wherein the system is installed on a server computer **and includes means operable** to collect and store events, process enterprise client signatures, generate events, operate a central system database, schedule events, initiate and respond to action requests, initiate self-healing counter-measures, maintain a graphical user interface backend structure, and manage mobile autonomous code agents.

46. **(Original)** The system according to claim 40, wherein the client computer has a capability to perform server computer functions in the event of failure of a server computer.

47. **(Original)** The system according to claim 40, wherein the server computer has a capability to perform client computer functions in the event of failure of a client computer.

48. **(Currently Amended)** Computer executable software code stored on a computer readable medium, the code for a computer implemented method for providing system security and resource management, comprising:

code for managing event messages by a master control processor between system handlers according to security system policies;

code for processing network messages by a network handler between client and server computers;

code for inserting native and third party event messages received by an insertion handler into the master control processor for processing by other system handlers;

code for detecting and processing event message signatures by the signature handler from alarm, system, and insertion events for conversion into system alarm messages for action by the other system handlers; and

code for performing actions by an action handler in response to action requests from the master control processor**, wherein the master control processor routes messages to and from the system handlers without being involved in security and resource management functions distributed among the system handlers**.

49. **(Original)** The computer executable software code method of claim 48, further comprising:

code for maintaining an execution environment by an agent handler for mobile autonomous code modules;

code for collecting and logging event messages by a logging handler; and

code for managing system configuration parameters by a configuration handler.